Санкт-Петербургский государственный университет
Прикладная математика и информатика
Статистическое моделирование

Иванова Елизавета Владимировна

Задачи анализа спектров тандемной масс-спектрометрии

Выпускная квалификационная работа

Научный руководитель:
к. ф.-м. н., доцент А. И. Коробейников

Рецензент:
разработчик ПО А. Л. Тарасов

Санкт-Петербург
2017

Saint Petersburg State University
Applied Mathematics and Computer Science
Statistical Modelling

Ivanova Elizaveta Vladimirovna

Problems of tandem mass-spectrometry spectra analysis

Graduation Project

Scientific Supervisor:
PhD, Associate Professor
Anton Korobeynikov

Reviewer:
Software Developer Artem Tarasov

Saint Petersburg
2017

Оглавление

Введение

Масс-спектрометрия — это техника, которую используют для определения химического состава веществ. Для исследуемого вещества метод строит масс-спектр — сигнал, который представляет собой зависимость интенсивности (количества) ионов от отношения массы к заряду иона.

Ввиду особенностей метода ионизации, взвешивания молекул, специфики прибора и самого вещества, получаемые масс-спектры вообще говоря зашумлены. Это влияет на результаты дальнейшего анализа, например, в методах идентификации вещества наличие шумовых пиков приводит к большому количеству ложных сопоставлений пептидам.

Так как масс-спектры активно используются во многих областях знаний, существует большое количество методов для удаления шума и других артефактов из масс-спектра.

Текущая процедура фильтрации пиков в Дерепликаторе [1] имеет ряд недостатков, самый важный из которых заключается в том, что в полученном спектре появляются новые пики.

Первая задача данной работы — исследовать существующие методы фильтрации пиков и выбрать наиболее подходящий из них или предложить модификацию для интегрирования в Дерепликатор.

Вторая задача связана с идентификацией пептидов. Существуют методы, которые решают задачу определения химической формулы пептида по масс-спектру, один из них использует базу данных пептидов. Для каждого пептида в базе данных специальным образом строится ожидаемый спектр, затем каждый спектр сравнивается с масс-спектром, для которого требуется узнать формулу пептида.

Если эмпирических масс-спектров много, то наивный алгоритм, требующий перебор всех пептидов из базы данных, работает достаточно медленно, но процедуру можно ускорить, кластеризовав или ожидаемые спектры, или эмпирические.

Таким образом, вторая задача данной работы — научиться кластеризовать эти спектры, подобрав подходящее векторное представление и меру близости между ними.

Далее кратко описан состав работы.

В главе 1 изложены базовые определения масс-спектрометрии и сформулированы задачи дипломной работы. В главе 2 помещены результаты исследования методов фильтрации пиков в масс-спектрах, их алгоритмы и сравнение.

Глава 3 содержит результаты, связанные с кластеризацией ожидаемых спектров. В ней описаны выбор оптимального векторного представления спектров, предложены варианты расстояния между спектрами и представлены результаты вычислительного эксперимента на реальных данных.

Глава 1

Определения и постановка задачи

В этом разделе изложены основные определения из масс-спектрометрии, а также подробно сформулированы две задачи данной работы — замена процедуры фильтрации пиков в Дерепликаторе и ускорение идентификации пептидов с помощью кластеризации масс-спектров.

1.1. Базовые определения масс-спектрометрии

Масс-спектрометрия — это метод исследования вещества, который основан на ионизации молекул вещества и последующем разделении образующихся ионов по их m/z — отношении массы иона к его заряду. Масс-спектрометрия используется в протеомике, геологии, физике, химии для идентификации структуры химических веществ и их количественного определения.

Распределения ионов по m/z называют *масс-спектрами*, а приборы, с помощью которых их получают, — *масс-спектрометрами*. Простейшее устройство масс-спектрометра включает себя источник ионизации, масс-анализатор (устройство для разделения ионов) и детектор.

Масс-спектрометр может иметь два масс-анализатора. Такой масс-спектрометр называют *тандемным*. Из разделенных в первом масс-анализаторе ионов выбираются те, которые представляют больший интерес, их разделяют на более мелкие фрагменты во втором масс-анализаторе, а результат сортируют по m/z. Такой подход используется в целевом анализе, то есть определении конкретных соединений в образцах.

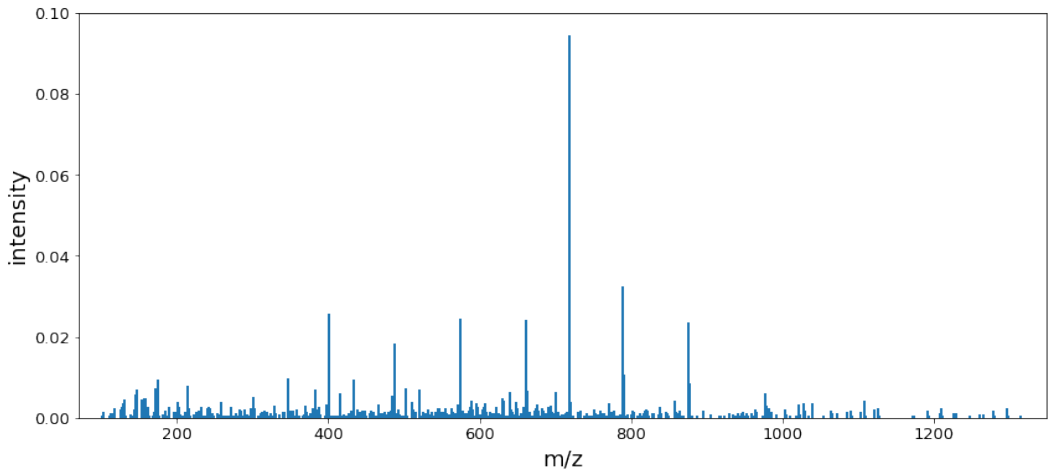


Рис. 1.1. Пример масс-спектра.

Для сокращения будем упускать слово «тандемный» перед масс-спектрами, но подразумевать именно такой тип технологии.

Пиками будем называть локальные максимумы в масс-спектре. Из-за особенностей метода ионизации, взвешивания молекул, специфики прибора и самого вещества спектры содержат шумовые пики, а их интенсивности искажены аддитивной медленно меняющейся компонентой — трендом (см. рисунок 1.2).

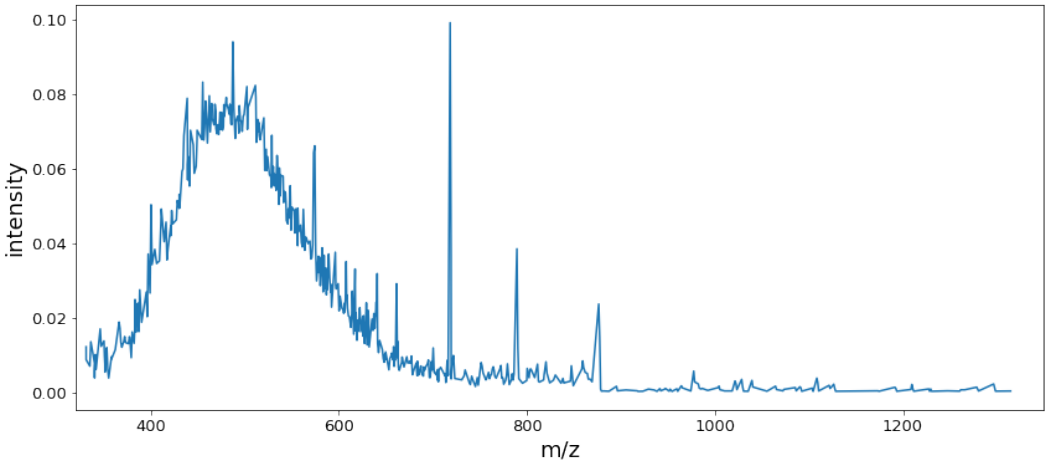


Рис. 1.2. Пример масс-спектра с выраженным трендом.

Так как это влияет на результаты дальнейшего анализа, появляется задача определить в спектре настоящие пики.

1.2. Дерепликатор и его алгоритм фильтрации пиков

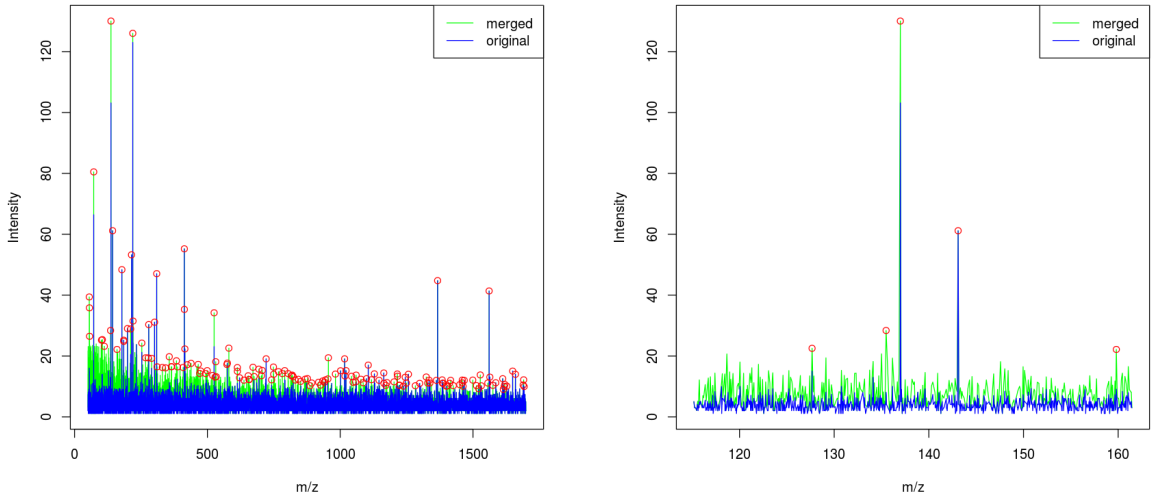
Дерепликатор (см. [1]) — это один из алгоритмов, который позволяет идентифицировать химическое вещество по масс-спектру. Зашумленность и наличие тренда в масс-спектрах приводит к увеличению доли ложных сопоставлений пептидам. Поэтому необходима процедура фильтрации пиков.

На текущий момент отбор пиков в Дерепликаторе выполняется в два шага — объединение и фильтрация. На первом шаге отсчеты m/z делятся на промежутки фиксированной длины, по умолчанию 50 Da, и в рамках каждого промежутка пики объединяются, если модуль разности их интенсивностей не превышается некоторого заданного порогового значения. На втором шаге в каждом окне отбираются пики с наибольшей интенсивностью, количество которых регулируется пользователем.

Этот алгоритм имеет ряд недостатков:

- Оба шага выполняются за квадратичное время от длины входа;
- Нет процедуры удаления тренда;
- Существенно увеличивается уровень шума;
- Появляются пики там, где в исходном спектре их не было.

Алгоритм идентификации в Дерепликаторе не учитывает значения интенсивностей, но преобразование на первом шаге отбора пиков сильно влияет на результаты второго.



(a) Весь спектр

(b) Участок спектра

Рис. 1.3. Пример результатов отбора пиков. Исходный спектр изображается синим цветом, спектр после объединения — зеленым цветом. Красные точки обозначают пики, которые осталась после фильтрации, именно они будут использоваться в дальнейшем анализе.

На рисунке 1.3 изображен спектр (a) и его подмножество (b). Видно, что после первого шага алгоритма интенсивности изменились непропорционально. На правом графике 3 из 5 пиков можно отнести к неправильно отфильтрованным, так как их исходные интенсивности сильно меньше преобразованных.

Оценим количество неправильно отобранных пиков во всем спектре следующим образом. Пусть $S = \{(m_k, i_k)\}_{k=1}^{n}$ — исходный спектр длины n, $\mathcal{F} \subset \{1, \ldots, n\}$ — индексы пиков, которые были отобраны двухшаговым препроцессингом, $S_{\mathcal{F}} = \{(m_k, i'_k)\}_{k \in \mathcal{F}}$ — соответствующий спектр. Для любого $k \geq 1$ введем

$$D_k(\delta) = \{\max(1, k - \delta), \ldots, \max(1, k - \delta/2)\} \cup \{\min(n, k + \delta/2), \ldots, \min(n, k + \delta)\}.$$

Вид такой «выколотой» окрестности обусловлен тем, что небольшой промежуток m/z может быть заполнен ненулевыми пиками, которые на самом деле соответствуют одному иону.

Для каждого пика $(m_k, i'_k) \in S_{\mathcal{F}}$ рассмотрим $\Delta_{(m_k, i_k)}(\delta) = \{(m_\ell, i_\ell)\}_{\ell \in D_k(\delta) \setminus \mathcal{F}}$. Будем считать пик (m_k, i'_k) сомнительным, если существует пик $(m_\ell, i_\ell) \in \Delta_{(m_k, i_k)}(\delta)$ такой, что $i_\ell > i_k$. Другими словами, в окрестности k-го пика в исходном спектре нашелся пик, у которого больше интенсивность и который после отбора пиков не принадлежит $S_{\mathcal{F}}$.

Таблица 1.1 содержит результаты для пяти спектров, первый из которых изображен на рисунке 1.3 (a). Заметим, что число подозрительных пиков во всех случаях превышает 30% от числа отобранных пиков. Таким образом, замена такого отбора пиков другим просто необходима.

Таблица 1.1. Результаты отбора пиков на примере пяти различных спектров. Параметры алгоритма: порог объединения — 0.05, длина окна — 50 Da, число отобранных пиков в рамках окна — 5.

Номер спектра	Число исходных пиков	Число пиков после шага 1	Число пиков после шага 2	Число сомнительных пиков
1	15547	11772	160	56
2	15296	11724	160	60
3	15218	11669	160	72
4	8736	7411	160	62
5	5813	5190	160	51

1.3. Об идентификации пептидов

1.3.1. Описание процедуры и модель ожидаемого спектра

Масс-спектрометрия является основным инструментом определения пептидов. Их идентификация важна для разработки новых лекарств, применяется в определении функций генов и во многих других задачах медицины и генетики.

Большинство методов идентификации пептидов по масс-спектру можно разделить на два типа — методы, использующие *базу данных пептидов*, и методы *de novo* секвенирования пептидов. В de novo секвенировании пептидную последовательность определяется исходя из разностей масс между соседними пиками. Идея этой стратегии заключается в том, что если разница между массой двух ионов равна массе одной аминокислоты, то скорее всего, эта аминокислота является частью пептидной последовательности. Этот метод подходит только для спектров с пренебрежительно малым уровнем шума. Поиск в базе данных тоже не лишен недостатков — идентификация зависит от того, насколько полная база пептидов используется.

Мы остановимся на первом способе идентификации пептидов, но прежде чем описать его схему, введем необходимые определения.

Рассмотрим алфавит аминокислот $\mathcal{A}$ размера 20 и функцию, сопоставляющую аминокислоте ее массу $m : \mathcal{A} \to (0, +\infty)$. Линейные пептиды, а именно такие пептиды мы будем рассматривать, представляют собой строку P из алфавита $\mathcal{A}$. Так как зачастую разбиение пептида в масс-спектрометре происходит вдоль него один раз, то на выходе получаем ионы, которые являются префиксами или суффиксами строки P. Их называют соответственно b- и y-ионами.

Определение 1.3.1. Пусть $P = p_0 p_1 \dots p_{n-1}$ — линейный пептид длины n, $p_i \in \mathcal{A}$. Ожидаемый спектр — это последовательность $\{b(P,k)\}_{k=1}^{n} \cup \{y(P,k)\}_{k=1}^{n}$, где

$$b(P,k) = \sum_{i=0}^{k-1} m(p_i),\ \ y(P,k) = \sum_{i=k}^{n-1} m(p_i)$$

— массы всех префиксов и суффиксов строки P.

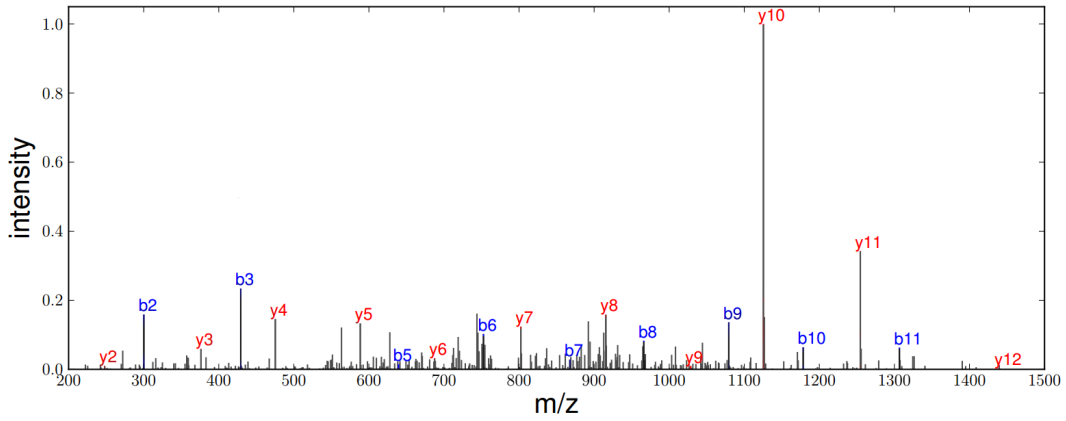


Рис. 1.4. Пример эмпирического масс-спектра с отмеченными b-, y-ионами, они изображены поверх наблюдаемых пиков синим и красным цветом соответственно. Оставшиеся пики — шумовые.

Следует отметить, что ожидаемые спектры, в отличие от эмпирических масс-спектров, — это лишь точки m/z, значит, для их сравнений не важны интенсивности.

Теперь можно описать схему идентификации пептидов с помощью базы данных. Пусть $\mathbf{P} = \{P_i\}_{i=1}^{M}$ — база данных пептидов размера M, $\mathcal{E}$ — функция, которая сопоставляет пептиду его ожидаемый спектр, $\mathcal{E}(\mathbf{P}) = \{\mathcal{E}(P_i)\}_{i=1}^{M}$ — множество ожидаемых спектров для $\mathbf{P}$. Пусть также $\mathbf{S} = \{S_j\}_{j=1}^{N}$ — множество эмпирических масс-спектров пептидов, для которых мы хотим определить химическую формулу.

Задачу идентификации можно записать так — при фиксированном τ для всех эмпирических масс-спектров $S_j \in \mathbf{S}$ найти $P_i \in \mathbf{P}$ такой, что $\text{score}(S_j, \mathcal{E}(P_i)) > \tau$, где score — некоторая оценка близости спектров, например, число общих пиков.

У этой задачи есть эквивалентная формулировка. При фиксированном τ для каждого пептида P_i нужно найти множество масс-спектров $\mathbf{M}_i \subset \mathbf{S}$, такое что

$$\text{score}(S_j, \mathcal{E}(P_i)) > \tau \text{ для каждого } \; S_j \in \mathbf{M}_i.$$

Эта задача имеет ряд сложностей, обусловленных спецификой данных. Во-первых, эмпирические масс-спектры зашумлены и в них отсутствуют некоторые b-, y-ионы. Во-вторых, позиции нешумовых пиков у масс-спектров, которые соответствуют одному пептиду, не обязательно совпадают, — это обязана учитывать оценка близости спектров. Например, пики будут считаться общими, если разница между их позициями по модулю не превосходит некоторого ε.

1.3.2. Варианты ускорения процедуры идентификации

Вернемся к постановке задачи идентификации и заметим, что наивный алгоритм, который для каждого P_i перебирает S_j, работает за $O(MN)$, где M — размер базы

данных, а N — число эмпирических масс-спектров. В случае огромной базы пептидов и огромного количества экспериментальных масс-спектров такой поиск затратен по времени.

Чтобы ускорить процедуру, можно кластеризовать как эмпирические спектры, так и ожидаемые.

Если кластеризовать эмпирические спектры, то получаем оценку времени работы $O(MK)$, где K — число кластеров. Если $N \gg M$ и $K \approx M$, то получим существенное ускорение процедуры.

Кластеризация ожидаемых спектров даст ускорение, если масс-спектры подаются на вход по одному. Если ответ на запрос для одного масс-спектра будет порядка $C \ll M$, то общее время работы $O(M^2 + NC)$ по сравнению с $O(NM)$.

В связи с этим возникает задача научиться кластеризовать экспериментальные и ожидаемые спектры, чтобы ускорить процедуру идентификации пептидов.

Глава 2

Исследование методов фильтрации масс-спектров

Большинство современных алгоритмов фильтрации масс-спектра можно разделить на три составляющие: сглаживание, удаление тренда и определение пиков [2].

Мы остановимся на описании алгоритмов PROcess [3] и MassSpecWavelet [4] из пакета Bioconductor в R, а также модификации MassSpecWavelet, которая реализована в msConvert из библиотеки ProteoWizard [5].

Эти алгоритмы имеют несколько параметров, которые легко подбираются, а с помощью параметра, который отвечает за минимальное значение отношения сигнала к шуму (signal to noise ratio threshold, SNR Th) можно контролировать количество отфильтрованных пиков. Алгоритмы из Bioconductor реализованы на R, а msConvert на C++, поэтому их просто тестировать для сравнения с текущим решением в Дерепликаторе.

2.1. Алгоритм PROcess

Здесь опишем детали алгоритма, который реализован в PROcess.

1. В первую очередь выполняется сглаживание спектра с помощью фильтра скользящего среднего для удаления шумовых пиков. Длина скользящего окна регулируется пользователем.

2. Удаление тренда происходит в два шага.

   a) Найти локальные минимумы в рамках скользящего окна.

   b) В каждой точке локального минимума подогнать локальную регрессию.

   Полученную кривую вычитают из спектра.

3. Оставшиеся пики фильтруются последовательно с помощью трех критериев:

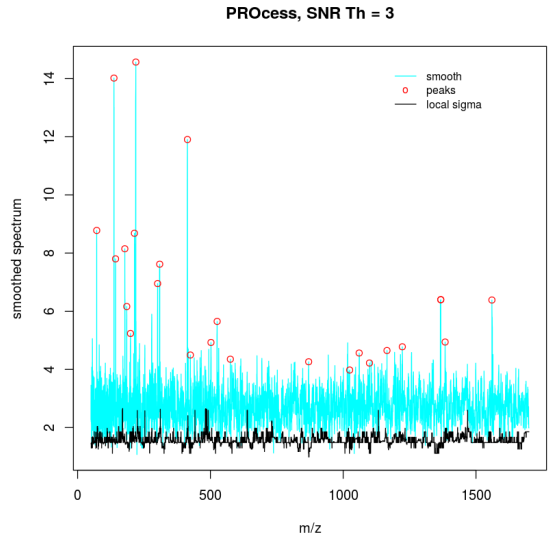
   a) Оценивается SNR как отношение сглаженного спектра к MAD (mean absolute deviation), посчитанной в каждом скользящем окне на первом шаге. Пик остается, если оценка SNR оказывается не меньше некоторого порогового значения.

   b) Пики, интенсивности которых меньше некоторого порога, удаляются.

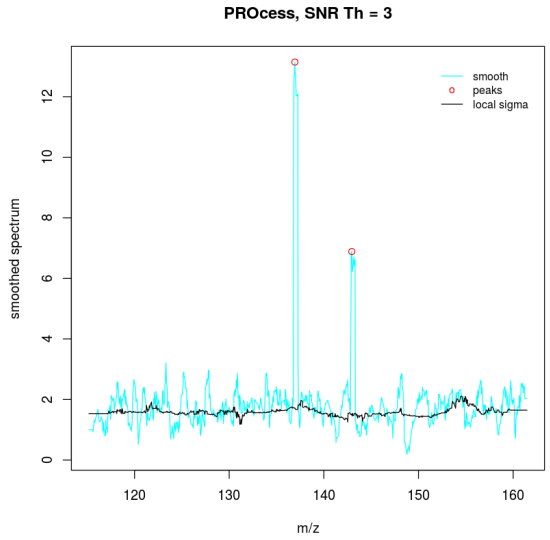
   c) Далее вычисляется площадь пика как площадь под кривой, соединяющий несколько пиков из окрестности текущего. Отбираются те пики, у которых отношение площади к максимуму площади по всем пикам больше некоторого значения.

На графиках рисунка 2.1 приведены результаты фильтрации спектра на рисунке 1.3 с помощью алгоритма PROcess при различных пороговых значениях для SNR.

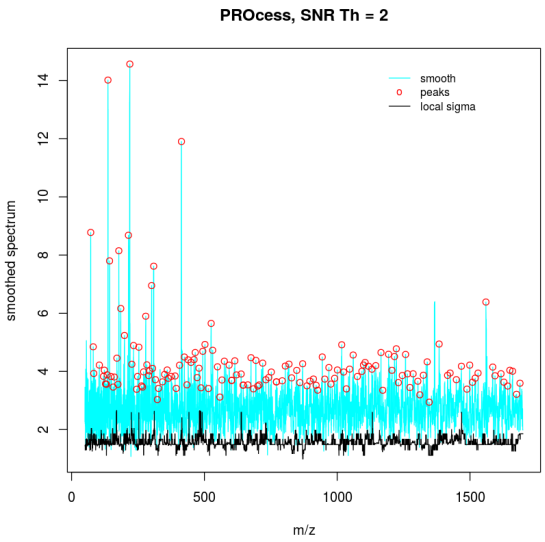
По сравнению с графиком 1.3 b) на графике 2.1 b) алгоритм отобрал два пика с наибольшими интенсивностями, что совпадает с нашими интуитивными представлениями о настоящих пиках в спектре.



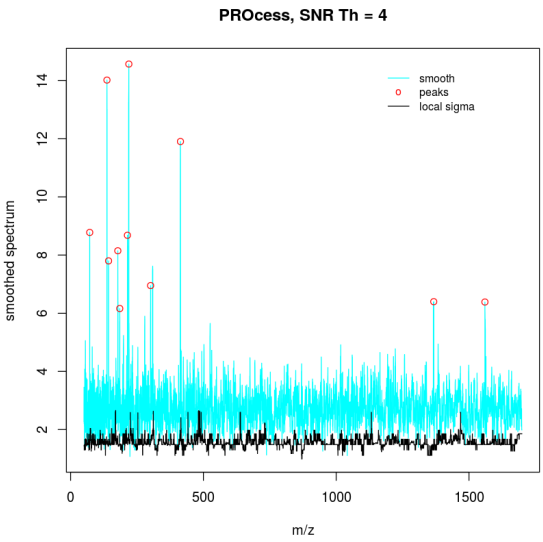
(a) Весь спектр, SNR Th = 3



(b) Участок спектра, SNR Th = 3



(c) Весь спектр, SNR Th = 2



(d) Весь спектр, SNR Th = 4

Рис. 2.1. Пример предобработки спектра с помощью PROcess.

2.2. Алгоритм MassSpecWavelet

Ключевая техника алгоритма в MassSpecWavelet — это непрерывное вейвлет-преобразование (Continuous Wavelet Transform, CWT). Идея алгоритма заключается в приближении каждого пика симметричной функцией с одним глобальным максимумом и параметром, отвечающим за масштаб.

Предположим, что исходный спектр удовлетворяет следующей модели

$$Y(t) = S(t) + B(t) + E(t),$$

где $S(t)$ — искомый спектр, $B(t)$ — тренд, $E(t)$ — шум с нулевым средним.

Пусть $\psi(t) \in L^2(\mathbb{R})$ — некоторая симметричная функция, удовлетворяющая условиям

$$\int\limits_{\mathbb{R}} \psi(t)dt = 0 \text{ и } \int\limits_{\mathbb{R}} \psi^2(t)dt = 1.$$

С помощью этой функции будем строить вейвлеты $\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right)$.

Посчитаем коэффициенты вейвлет-преобразования для $Y(t)$

$$\int\limits_{\mathbb{R}} \psi_{a,b}(t)Y(t)dt = \int\limits_{\mathbb{R}} \psi_{a,b}(t)S(t)dt + \int\limits_{\mathbb{R}} \psi_{a,b}(t)B(t)dt + \int\limits_{\mathbb{R}} \psi_{a,b}(t)E(t)dt.$$

Если тренд монотонен в окрестности пика, то его можно аппроксимировать некоторой нечетной функцией в этой окрестности. Следовательно, второй член разложения примерно равен нулю, так как ψ симметрична с нулевым средним. Третьим членом разложения тоже можно пренебречь, так как шум имеет нулевое среднее.

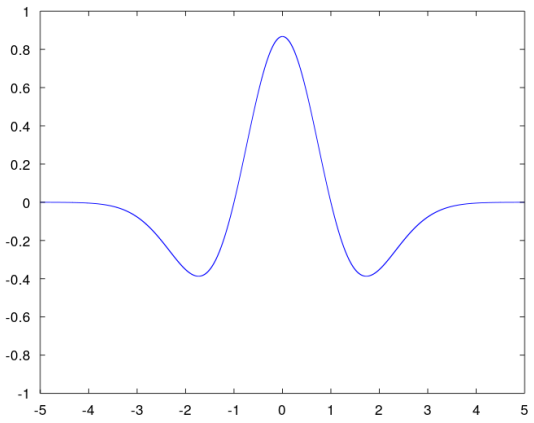


Рис. 2.2. Вейвлет Рикера

Таким образом, при использовании вейвлет-преобразования нет необходимости отдельно оценивать и удалять шум и тренд. Получаем преимущество перед PROcess, так как здесь нет параметров, с помощью которых регулируется сглаживание и оценивание тренда, что делает процедуру более устойчивой.

Обычно в качестве базового вейвлета ψ берут вторую производную гауссианы с противоположным знаком. Этот вейвлет называют «мексиканская шляпа» (mexican hat) или вейвлет Рикера (Ricker wavelet), его график изображен на рисунке 2.2. Его формула

$$\psi(t) = \frac{2}{\sqrt{3}\pi^{1/4}}\left(1 - t^2\right)e^{-t^2/2}.$$

Особенность этого вейвлета по сравнению с другими симметричными функциями с одним глобальным максимумом в наличии глобальных минимумов, которые позволяют точнее подогнать функцию к форме пика.

Опишем алгоритм идентификации пиков в MassSpecWavelet.

1. Фиксируется наибольшее значение масштаба a, которое обозначим за A. Для $a = 1, \ldots, A$ подсчитываются коэффициенты вейвлет-преобразования с помощью свертки, затем в полученном ряде находятся локальные максимумы в рамках скользящего окна, длина которого пропорциональна параметру a (в реализации равна $2a + 1$).

2. Локальные максимумы соединяются в гребни (ridges) следующим образом. Пусть $M_{\mathrm{CWT}} \in \mathbb{R}^{m \times n}$ — матрица вейвлет-коэффициентов, где каждая строка соответствует определенному масштабу a. Тогда в цикле по $i = m, \ldots, 2$ для каждого локального максимума в $M_{\mathrm{CWT}}[i, :]$, индекс которого обозначим за k, находим в $M_{\mathrm{CWT}}[i+1, k-a : k+a]$ локальный максимум. Если такой нашелся, то добавляем его в k-й гребень, иначе увеличиваем k-й счетчик пропусков, который обнуляется каждый раз, когда находится следующий локальный максимум. Гребень строится, пока не его счетчик пропусков не превысит некоторое пороговое значение.

3. Определяется SNR. Сигнал оценивается как максимальный вейвлет-коэффициент в рамках гребня в пределах некоторого значения масштаба a. Для оценки шума используется 95-процентный квантиль от модуля вейвлет-коэффициентов для $a = 1$ в рамках скользящего окна, длина которого регулируется пользователем.

4. Пики отбираются так, чтобы они удовлетворяли следующим условиям:

   a) Масштаб, который соответствует максимальному коэффициенту в гребне, с помощью которого мы оцениваем сигнал, должен принадлежать определенному промежутку, который задается пользователем.

   b) SNR не должен быть меньше определенного порогового значения.

   c) Длина гребня должна быть больше заданной.

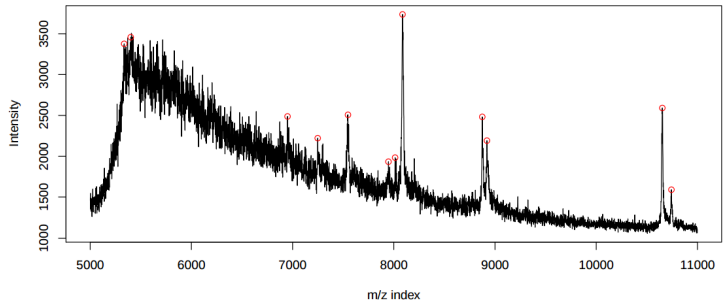
На рисунке 2.3 изображены графики, иллюстрирующие соответствие между пиками и локальными максимумами вейвлет-коэффициентов и показывающие, как выглядят гребни, построенные по этим коэффициентам.

На рисунке 2.4 приведены результаты обработки спектра 1.3 с помощью MassSpecWavelet при различных значениях параметра SNR Th.

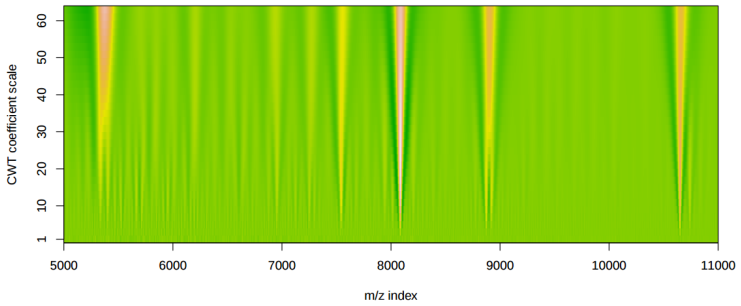
Заметим, что описанный алгоритм в статье [4] несколько отличается от реализации на R. Во первых, на первом шаге локальные максимумы ищутся не в рамках скользящего окна с шагом 1 (как принято, если не оговаривается обратное), а с шагом, равным половине длины окна, и кроме того, только в рамках окна выбирается только один

максимум. Во-вторых, пики на границах спектра отбрасываются. Такие модификации были предприняты с целью сделать процедуру отбора пиков более устойчивой.

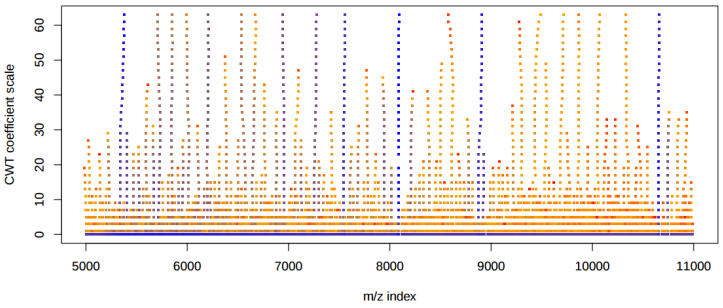
После нескольких экспериментов алгоритм MassSpecWavelet был выбран в качестве альтернативы текущему препроцессингу в Дерепликаторе, он использует немного параметров, и они легко настраиваются, а удаление шума и тренда происходит автоматически.



(a) Исходный спектр. Отобранные пики отмечены красными окружностями

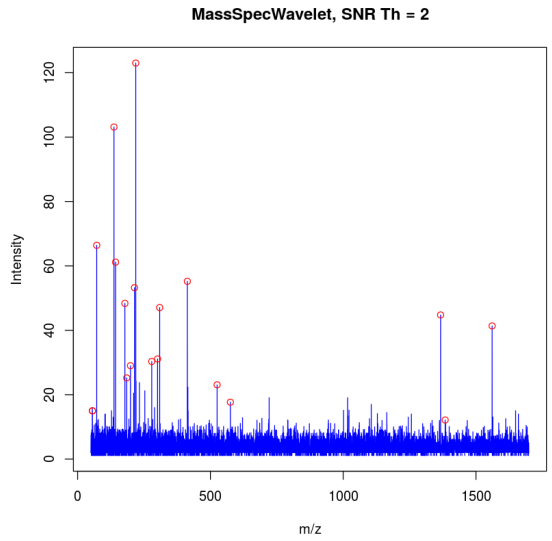


(b) Коэффициенты вейвлет-преобразования для $a = 1, \ldots, 64$. Зеленые соответствуют минимальным значениям, красные — максимальным

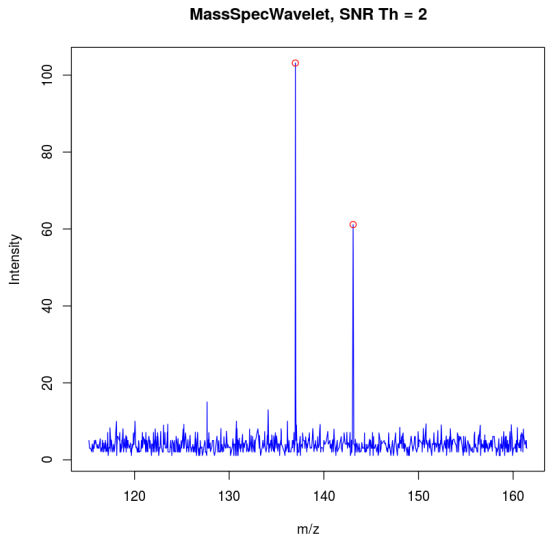


(c) Локальные максимумы таблицы коэффициентов, соединенные в гребни

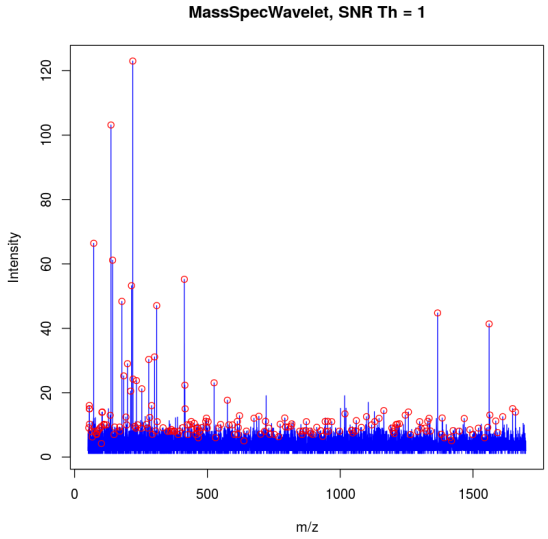
Рис. 2.3. Определение пиков с помощью непрерывного вейвлет-преобразования



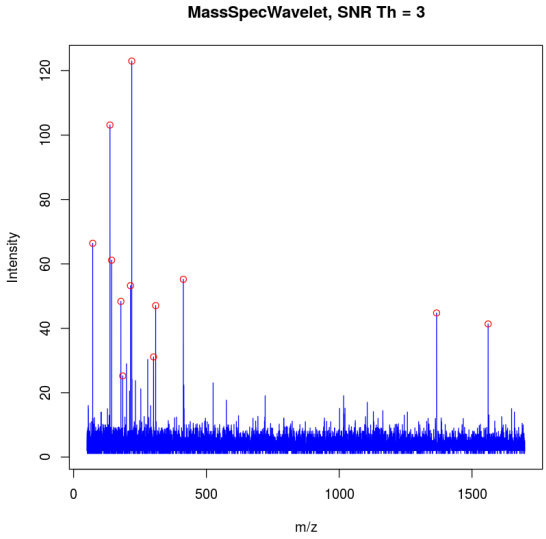
(a) Весь спектр, SNR Th = 2



(b) Участок спектра, SNR Th = 2



(c) Весь спектр, SNR Th = 1



(d) Весь спектр, SNR Th = 3

Рис. 2.4. Пример предобработки спектра с помощью MassSpecWavelet

Однако алгоритм неявно предполагает, что m/z расположены на равном расстоянии друг от друга. В реальности же промежутки между m/z не равны, а приведение спектра к необходимому виду путем добавления пиков с интенсивностью 0 приведет к увеличению времени работы алгоритма. Поэтому хотелось бы использовать модификацию, которая каким-нибудь разумным образом учитывала распределение промежутков между m/z.

2.3. Алгоритм msConvert из библиотеки ProteoWizard

В библиотеке ProteoWizard реализован инструмент для определения пиков, который использует идею MassSpecWavelet, но его алгоритм подходит и для не равноотстоящих m/z. Его идея заключается в том, чтобы сделать диапазон масштаба a, для которого считаются вейвлет-коэффициенты, зависимым от приращений m/z.

Кратко изложим алгоритм, который предложен в [5].

1. Пусть $m \in \mathbb{R}^n_+$ — вектор m/z для спектра длины n. Для каждого $i = 1, \ldots, n$ посчитаем

$$s[i] = \sum_{j=\max(1,i-\ell)}^{\min(n,i+\ell)-1} \frac{m[i+1] - m[i]}{2\ell},$$

   и в реализации выбрано $\ell = 5$. Вейвлет-коэффициенты будут рассчитываться для вейвлетов, центрированных в $m[i]$ и точках $(m[i]+m[i+1])/2$ при $a = s[i], \ldots, 7s[i]$.

2. Обозначим за $M_{\mathrm{CWT}} \in \mathbb{R}^{7\times 2n-1}$ матрицу вейвлет-коэффициентов. m/z делятся на бины по r точек, для каждого рассчитывается 95-процентный квантиль коэффициентов, соответствующих наименьшему масштабу, — это и будет оценкой шума:

$$e[k] = q_{0.95}\big(M_{\mathrm{CWT}}[1, k : \min(2n-1, k+2r-1)]\big), \text{ где } k = 1, \ldots, (2\lfloor n/r \rfloor + 1).$$

   Для $i = 1, \ldots, 2n-1$ оценивается SNR:

$$\mathrm{SNR}[i] = \max_j M_{\mathrm{CWT}}[j, i]/e[k], \text{ где } k \le i \le \min(2n-1, k+2r-1)].$$

   В программной реализации r выбрано равным 300.

3. Отбираются пики, для которых SNR больше некоторого порога, и если среди них есть пики, m/z которых отличаются на 0.1, то удаляются те, у которых меньше SNR.

Несмотря на то, что процедура отбора пиков здесь менее строгая, алгоритм хорошо работает на практике для длинных спектров. Из недостатков этой реализации нужно отметить обилие относительных значений параметров, а не абсолютных — m, ℓ и другие. Как следствие, алгоритм работает некорректно на спектрах малой длины.

В рамках этой работы реализация из msConvert была исправлена, все относительные параметры в ней вычисляются исходя из длины спектра или задаются пользователем.

2.3.1. Вычислительные эксперименты

В этом параграфе приводятся результаты вычислительных экспериментов, цель которых сравнить фильтрацию из Дерепликатора и фильтрацию из msConvert.

В первую очередь вернемся к данным таблицы 1.1 и дополним ее результатами для реализации, основанной на алгоритме из msConvert. Объединенные результаты показаны в таблице 2.2. Можно заметить, что число пиков сомнительных пиков после фильтрации msConvert, составляет в среднем около 21% вместо 38% в случае Дерепликатора (с дисперсией около 3 и 5 соответственно).

Таблица 2.1. Сравнение фильтрации спектров в Дерепликаторе и msConvert. Параметры первого алгоритма взяты из таблицы 1.1, параметры второго алгоритма: SNRTh = 1.5.

Номер спектра	Длина спектра	Дерепликатор		msConvert	
		Число пиков после фильтрации	Число сомнительных пиков	Число пиков после фильтрации	Число сомнительных пиков
1	15547	160	56	151	30
2	15296	160	60	123	25
3	15218	160	72	126	33
4	8736	160	62	56	12
5	5813	160	51	33	6

Таблица 2.2. Сравнение фильтрации спектров в Дерепликаторе и msConvert. Параметры первого алгоритма: порог объединения — 0.05, длина окна — 50 Da, число отобранных пиков в рамках окна — 2. Параметры второго алгоритма: SNRTh = 0.6.

Номер спектра	Длина спектра	Дерепликатор		msConvert	
		Число пиков после фильтрации	Число сомнительных пиков	Число пиков после фильтрации	Число сомнительных пиков
1	692	48	2	49	1
2	642	47	3	44	1
3	660	54	4	50	2
4	654	55	3	46	2
5	600	55	4	50	0
6	617	48	4	41	3

Теперь рассмотрим другой набор данных — множество спектров, для каждого из которых был идентифицирован пептид с некоторой малой погрешностью. Длины спектров в данных варьируются от 200 до 1500.

Эксперименты над спектрами с известными пептидами показали, что оба метода фильтрации (при некоторых заранее подобранных параметрах) оставляют достаточное количество значимых пиков, чтобы подтвердить пептид, к которому относится масс-спектр.

Опишем детальнее один из таких экспериментов. Пусть $\{S_i\}_{i=1}^n$ — масс-спектры, $\{P_i\}_{i=1}^n$ — соответствующие им пептиды, а $f(S_i, P_j) = f_\varepsilon(S_i, P_j)$ — число пиков S_i, m/z которых отличается от позиций y-,b-ионов ожидаемого спектра для P_j не больше, чем на ε. Будем называть такие пики значимыми. Для краткости $f(S_i) := f(S_i, P_i)$.

Пусть также $g(S_i) = g_\varepsilon(S_i) = \sum_{P_i \neq P_j} f_\varepsilon(S_i, P_j) / \sum_{i=1}^n \mathbb{I}\{P_i \neq P_j\}$ — среднее число значимых пиков между масс-спектром S_i и ожидаемыми спектрами пептидов, отличных от P_i. Через $\widehat{S}_i$ обозначим отфильтрованный спектр, а через $|S_i|$ длину спектра.

Для представления результатов были выбраны спектры длиной от 600 до 700. В таблицах 2.2 и 2.3 приведены результаты их фильтрации. Значимые пики в эксперименте определялись согласно $\varepsilon = 5 \cdot 10^{-3}$.

Таблица 2.3. Сравнение фильтрации спектров в Дерепликаторе и msConvert на масс-спектрах с известными пептидами. Параметры алгоритмов аналогичны параметрам из таблицы 2.2.

i	$f(S_i)$	$f(S_i)/\lvert S_i\rvert$	$g(S_i)$	Дерепликатор			msConvert		
				$f(\widehat{S}_i)$	$f(\widehat{S}_i)/\lvert\widehat{S}_i\rvert$	$g(\widehat{S}_i)$	$f(\widehat{S}_i, P_i)$	$f(\widehat{S}_i)/\lvert\widehat{S}_i\rvert$	$g(\widehat{S}_i)$
1	29	0.04	2.24	15	0.31	0.15	15	0.31	0.21
2	24	0.05	2.95	14	0.29	0.33	11	0.25	0.34
3	29	0.04	3.37	19	0.35	0.36	19	0.38	0.29
4	29	0.04	3.13	19	0.34	0.30	20	0.43	0.36
5	30	0.05	2.96	16	0.29	0.41	20	0.4	0.37
6	25	0.04	3.45	13	0.27	1.20	19	0.46	1.14

Видно, что для обоих методов фильтрации доля значимых пиков в спектрах возросла, среднее число значимых пиков для пептидов, отличных от данного тоже уменьшилось. MsConvert оставляет достаточное количество значимых пиков для идентификации, но его результаты сходны с результатами фильтрации в Дерепликаторе.

Исходя из таблиц 2.1 и 2.2, а именно того, что для больших сильно зашумленных спектров процедура фильтрации в Дерепликаторе оставляет в два раза больше подозрительных пиков, чем msConvert, можно предположить, что использование msConvert будет приводить к более точным результатам. Однако данные, которые могут подтвердить это утверждение, сложно найти.

Глава 3

Результаты кластеризации масс-спектров

Кластеризация эмпирических спектров гораздо сложнее, чем ожидаемых, так как эмпирические масс-спектры зашумлены, в них отсутствуют некоторые b-, y-ионы, а позиции нешумовых пиков у масс-спектров, которые соответствуют одному пептиду, могут не совпадать. Поэтому остановимся на кластеризации ожидаемых масс-спектров. В данном разделе будет помещены описания проделанных исследований и результаты вычислительных экспериментов.

3.1. Кластеризация ожидаемых масс-спектров

Задача кластеризации будет включать в себя несколько пунктов:

1. Выбор расстояния между пептидами как строками.
2. Выбор подходящего представление масс-спектра, так как масс-спектры не являются элементами векторного пространства.
3. Выбор расстояния между масс-спектрами (точнее, между их векторными вложениями), которая была бы согласована с расстоянием между пептидами и вычислялась за разумное время.
4. Выбор алгоритма кластеризации.

Специфика кластеризации ожидаемых спектров заключается в условии — если близки спектры, то близки и строки-пептиды, которые им соответствуют. То есть для некоторого малого ε, если $\text{dist}(\mathcal{E}(P_i), \mathcal{E}(P_j)) < \varepsilon$, то $\text{stringDist}(P_i, P_j) < \delta(\varepsilon)$, где $P_i, P_j \in \mathbf{P}$ — базы данных пептидов, $\mathcal{E}$ — функция, которая сопоставляет пептиду его ожидаемый спектр, dist — некоторое расстояние между спектрами, stringDist — некоторое расстояние между строками. Но на практике может оказаться, что близким спектрам соответствуют совсем разные пептиды, поэтому будем искать расстояние, при котором число таких несовпадений относительно мало.

3.1.1. Об оценке близости пептидов как строк

Прежде, чем предложить способ измерения похожести строк, введем понятие выравнивания строк.

Отступление о выравнивании строк

Пусть $S_1 = s_1^{(1)}, \ldots, s_n^{(1)}$ и $S_1 = s_1^{(2)}, \ldots, s_m^{(2)}$ — две строки над некоторым алфавитом $\mathcal{A}$, то есть $s_i^{(1)}$, $s_j^{(2)} \in \mathcal{A}$, $i = 1, \ldots, n$, $j = 1, \ldots, m$. *Выравниваем* будем называть пару

(S'_1, S'_2) такую, что

$$S'_1 = g^{(1)}_{0,1} s^{(1)}_1 \dots g^{(1)}_{n-1,n} s^{(1)}_n g^{(1)}_{n,n+1},\ S'_2 = g^{(2)}_{0,1} s^{(2)}_1 \dots g^{(1)}_{m-1,m} s^{(1)}_m g^{(1)}_{m,m+1},$$

где $g^{(i)}_{k,\ell}$ равно пустому символу или последовательности пробелов некоторой длины $t_{k,\ell,i} > 0$, а длины строк S'_1 и S'_2 совпадают. Для обозначения пробелов будем использовать символ «_», считая, что он не принадлежит алфавиту $\mathcal{A}$.

Пару (S'_1, S'_2) можно представить как последовательность пар символов, где каждая пара относится к одному из трех случаев $(\_, s^{(2)}_j)$, $(s^{(1)}_i, \_)$, $(s^{(1)}_i, s^{(2)}_j)$. Первых два случая соответствуют *пропускам* в одной из строк. В последнем случае, если $s^{(1)}_i \neq s^{(2)}_j$, будем говорить, что произошла *замена* $s^{(1)}_i$ на $s^{(2)}_j$, воспринимая строку S_1 как строку перед модификацией, а S_2 — после модификации.

Из всем возможных (S'_1, S'_2) нас интересуют те, у которых число пар $(s^{(1)}_i, s^{(2)}_j)$ с $s^{(1)}_i = s^{(2)}_j$ наибольшее, так как оно в некотором смысле отражает близость строк S_1, S_2. Формально измерить близость строк (S_1, S_2) по (S'_1, S'_2) позволяет *матрица замены* и *штрафы за пропуски*. Матрица замены — это квадратная матрица, размер которой равен мощности алфавита $\mathcal{A}$ и каждый элемент которой описывает частоту замен одного символа другим. Пример такой матрицы приведен на рисунке 3.1.

Что касается пропусков, то для них задается отдельная функция штрафов. Она может быть линейной — $w(g^{(i)}_{k,\ell}) = -ht_{k,\ell,i}$ — или аффинной — $w(g^{(i)}_{k,\ell}) = -h - u(t_{k,\ell,i} - 1)$ при $t_{k,\ell,i} > 1$ и $w(g^{(i)}_{k,\ell}) = -h$ при $t_{k,\ell,i} = 1$, $h > u$. Основное отличие этих функций в том, что вторая функция присваивает меньший штраф за подряд идущие пробелы.

Обозначим через b длину $S'_1 = s'^{(1)}_1, \dots, s'^{(1)}_b$ и $S'_2 = s'^{(2)}_1, \dots, s'^{(2)}_b$, а через $\mathbf{X}$ — матрицу замены, для простоты будем считать, что ее индексы это литеры алфавита $\mathcal{A}$. Таким образом можно ввести *оценку выравнивания* в случае линейных штрафов

$$\text{score}(S'_1, S'_2) = \sum_{k=1}^{b} \begin{cases} \mathbf{X}[s^{(1)}_i, s^{(2)}_j], & \text{если } (s'^{(1)}_i, s'^{(2)}_j) = (s^{(1)}_i, s^{(2)}_j), \\ -h, & \text{иначе.} \end{cases}$$

В случае аффинных штрафов оценка выравнивания отличается тем, что за продолжение последовательности пробелов дается штраф $-u$ вместо $-h$.

Оптимальным выравниванием будем называть пару (S'_1, S'_2), максимизирующую $\text{score}(S'_1, S'_2)$. Значение оценки оптимального выравнивания можно использовать как оценку близости строк S_1, S_2.

Еще несколько вспомогательных фактов о выравниваниях.

1) Выравнивание строк может быть глобальным или локальным. Случай глобального выравнивания рассмотрен выше. От локального оно отличается тем, что пропуски в начале и конце строк не учитываются.

2) Был описан случай парного выравнивания. Существует также множественное выравнивание для числа строк больше 2. На его описании мы не будем останавливаться, лишь отметим, что элементы каждой строки в таком выравнивании можно условно разделить на вариабельные и консервативные участки.

Оценка близости строк-пептидов

Особенность строк-пептидов в том, что некоторые аминокислоты связаны между собой, или в мутациях вероятность некоторых замен выше, чем других. Для строк-пептидов существуют матрицы замены, которые учитывают эти особенности, например, семейство матриц BLOSUM (BLOck SUbstitution Matrix, [6]).

	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	
C	9																				C
S	-1	4																			S
T	-1	1	5																		T
P	-3	-1	-1	7																	P
A	0	1	0	-1	4																A
G	-3	0	-2	-2	0	6															G
N	-3	1	0	-2	-2	0	6														N
D	-3	0	-1	-1	-2	-1	1	6													D
E	-4	0	-1	-1	-1	-2	0	2	5												E
Q	-3	0	-1	-1	-1	-2	0	0	2	5											Q
H	-3	-1	-2	-2	-2	-2	1	-1	0	0	8										H
R	-3	-1	-1	-2	-1	-2	0	-2	0	1	0	5									R
K	-3	0	-1	-1	-1	-2	0	-1	1	1	-1	2	5								K
M	-1	-1	-1	-2	-1	-3	-2	-3	-2	0	-2	-1	-1	5							M
I	-1	-2	-1	-3	-1	-4	-3	-3	-3	-3	-3	-3	-3	1	4						I
L	-1	-2	-1	-3	-1	-4	-3	-4	-3	-2	-3	-2	-2	2	2	4					L
V	-1	-2	0	-2	0	-3	-3	-3	-2	-2	-3	-3	-2	1	3	1	4				V
F	-2	-2	-2	-4	-2	-3	-3	-3	-3	-3	-1	-3	-3	0	0	0	-1	6			F
Y	-2	-2	-2	-3	-2	-3	-2	-3	-2	-1	2	-2	-2	-1	-1	-1	-1	3	7		Y
W	-2	-3	-2	-4	-3	-2	-4	-4	-3	-2	-2	-3	-3	-1	-3	-2	-3	1	2	11	W
	C	S	T	P	A	G	N	D	E	Q	H	R	K	M	I	L	V	F	Y	W	

Рис. 3.1. Матрица замены BLOSUM62.

Элементы матрицы замены вычисляются по «блокам» — консервативным регионам множественного локального выравнивания набора белков из базы данных Blocks, подробнее в [6].

Элементы матрицы замены вычисляются как целая часть выражения

$$s(a,b) = \frac{1}{\sqrt{\lambda}} \log \frac{\hat{p}_{ab}}{\hat{q}_a \hat{q}_b} \ ,$$

где $a, b \in \mathcal{A}$ — аминокислоты, $\hat{p}_{ab}$ — доля замен a на b, $\hat{q}_a, \hat{q}_b$ — доля аминокислот a и b в блоках, а λ — коэффициент масштабирования, который выбирается так, чтобы целочисленные значения $s(a,b)$ были различны.

Матрицы в семействе отличаются строками, по которым оцениваются их параметры. Это отличие зашифровано в названии — BLOSUMr значит, что матрица построена по последовательностям с не более, чем r%-ным сходством между собой. Матрицы BLOSUM с большим r предназначены для сравнения тесно связанных последовательностей, а матрицы с небольшим r — для сравнения отдаленно связанных последовательностей. Матрица BLOSUM62, приведенная на рисунке 3.1, считается наилучшей в обнаружении слабых похожестей между пептидами, она используется по умолчанию во многих программах, выравнивающих белки. Именно ее будем использовать для оценивания близости пептидов.

При локальном выравнивании строка-пептид и все пептиды, являющиеся ее подстроками, будут лежать в одном кластере, но разница в длине между ними может быть

сколь угодно большой, поэтому мы остановимся на глобальном выравнивании.

Теперь, если $\mathrm{bs}(P_i, P_j)$ — это оценка глобального выравнивания пептидов P_i и P_j с матрицей замены BLOSUM62 и аффинными штрафами за пропуски с $h = 7$ и $u = 5$. Будем использовать следующую величину для измерения близости пептидов-строк:

$$\mathrm{stringMatchingScore}(P_i, P_j) := \frac{\mathrm{bs}(P_i, P_j)}{\sqrt{\mathrm{bs}(P_i, P_i)\,\mathrm{bs}(P_j, P_j)}} \ .$$

Такая величина по модулю не превосходит 1 и достигает своего максимума на совпадающих строках пептидах.

3.1.2. О векторном вложении масс-спектров

Ожидаемые спектры можно воспринимать как пары $(m/z, 1)$, то есть как дискретную меру на $(0, +\infty)$ с весами атомов, равными 1. Наивный способ векторного вложения M ожидаемых спектров — привести все меры, соответствующие им, к одному носителю с равноотстоящими отсчетами, дополнив веса нулями. Так как носитель равноотстоящий, то меры можно однозначным образом представить в виде векторов из нулей и единиц, однако такие вектора будут огромной размерности — если точность m/z достигает тысячных долей, а максимальный m/z порядка 1000, то получаем вектора размера 10^6.

Другой способ, более экономный с точки зрения занимаемой памяти, это представить спектры в виде гистограмм, ширину столбцов которых можно настраивать. Графическая иллюстрация такого представления изображена на рисунке 3.2. Именно такое векторное вложение спектров будет использоваться для кластеризации.

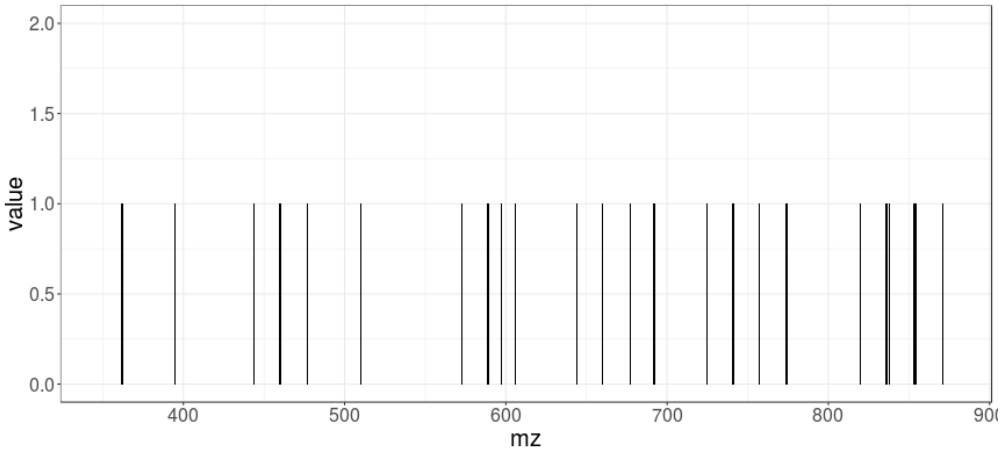
3.1.3. О расстояниях между масс-спектрами

Мы рассмотрим три способа измерения похожести спектров. Пусть $R = (r_0, \ldots, r_{n-1})$ и $Q = (q_0, \ldots, q_{n-1})$, где $r_i,\ p_i \in [0, +\infty)$ — две гистограммы, построенные по ожидаемым спектрам с некоторой шириной столбца.

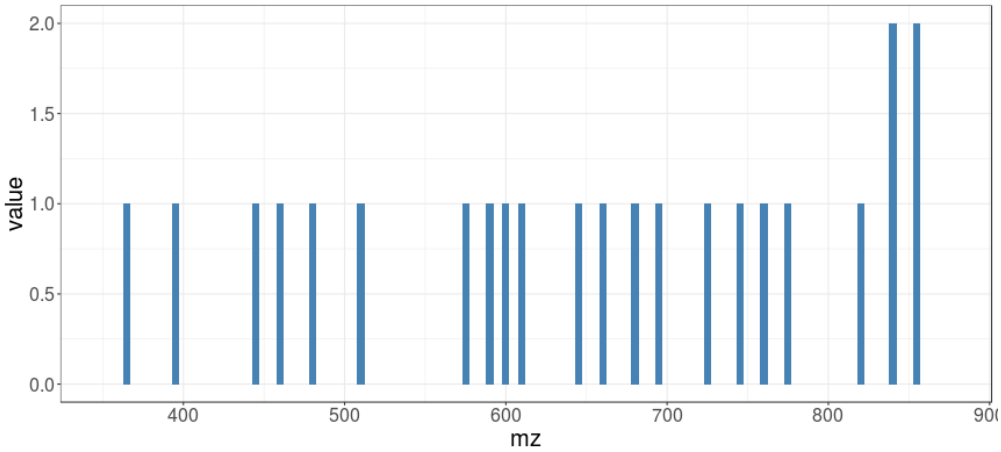
1. Синус угла между векторами-гистограммами:

$$\mathrm{SIN}(R, Q) = \sqrt{1 - \frac{\langle R, Q\rangle^2}{\|R\|_2^2 \|Q\|_2^2}} \ .$$

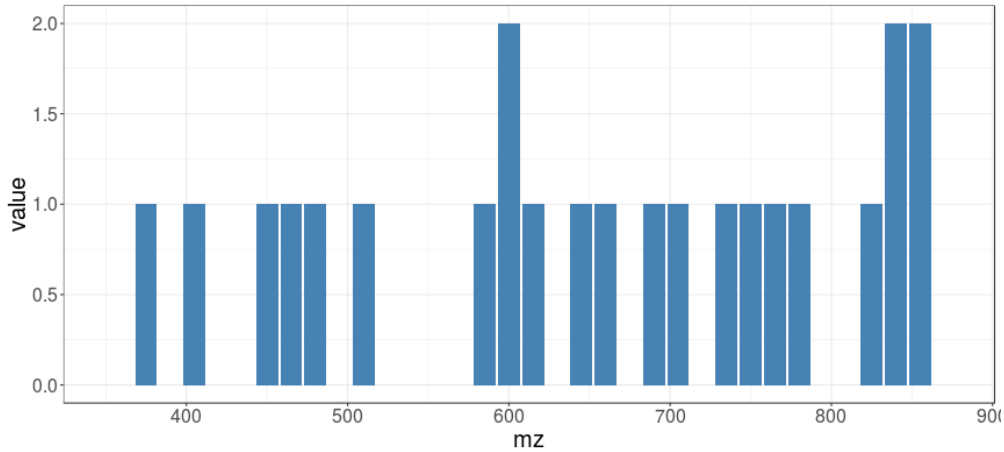
Ясно, что временная сложность его подсчета $O(n)$.



(a) Исходный спектр



(b) bin length = 5



(c) bin length = 15

Рис. 3.2. Пример представления спектра в виде гистограмм с шириной столбца bin length.

2. Метрика Васерштейна (Earth Mover's Distance) — метрика в пространстве вероятностных мер. Пусть $d_{ij} = \text{dist}(i, j)$ — некоторое расстояние между i-м столбцом гистограммы R и j-м столбцом гистограммы Q, тогда метрика вычисляется как

решение оптимизационной задачи (см. [7]):

$$\mathrm{EMD}(R,Q) = \min_{f_{ij}} \sum_{ij} f_{ij} d_{ij} / \sum_{ij} f_{ij} \;\; s.t. \;\; \sum_{ij} f_{ij} = \min\big(\sum_i r_i, \sum_j q_j\big), \tag{3.1.1}$$

$$\sum_j f_{ij} \le r_i,\ \sum_i f_{ij} \le q_j,\ f_{ij} \ge 0. \tag{3.1.2}$$

Смысл этой метрики прост — это минимальная стоимость преобразования одной гистограммы в другую с учетом расстояния d_{ij} между всевозможными атомами i и j.

Если $d_{ij} = |i-j|$, а R и Q — нормализованы, то EMD вычисляется достаточно просто за линейное время. На практике востребован вариант $d_{ij} = \min(|i-j|,\, t)$, который устойчив к выбросам.

Однако оптимизационная задача с таким видом d_{ij} решается с помощью частного случая алгоритма линейного программирования, которому в худшем случае требуется экспоненциальное время, но на практике время работы суперкубическое, то есть между $O(n^3)$ и $O(n^4)$, подробнее в [7].

Кроме того, EMD является метрикой лишь для лишь нормализованных гистограмм, а нормализация может привести к потере точности.

В статье [8] был предложен следующий вариант оптимизационной задачи:

$$\widehat{\mathrm{EMD}}(P,Q) = \min_{f_{ij}} \sum_{ij} f_{ij} d_{ij} + \big|\sum_i r_i - \sum_j q_j\big| \max_{ij} d_{ij} \tag{3.1.3}$$

при тех же ограничениях, что и в (3.1.1). Для нормализованных гистограмм задачи (3.1.1) и (3.1.3) эквивалентны. В [8] доказан теоретический факт, что если dist — это метрика, то $\widehat{\mathrm{EMD}}$ — метрика, в том числе для ненормализованных гистограмм.

В [9] для случая $d_{ij} = \min(|i-j|,\, t)$ предложено свести эту задачу к нахождению потока минимальной стоимости. В статье также говорится, что что их реализация, доступная для скачивания, работает за $O(\min(t^2 m, m^2))$, где m — суммарное число ненулевых столбцов гистограмм.

Именно эта программная реализация будет использована в экспериментах, и далее под аббревиатурой EMD будет пониматься решение оптимизационной задачи (3.1.3).

3. Quadratic-Chi Histogram Distance Family (см. [10]).

Пусть U — верхняя граница r_i, q_i. Рассмотрим $A \in [0,U]^{n\times n}$, $A_{ii} > A_{ij} = A_{ji}$ для всех i, j и некоторое $0 \le b < 1$. Тогда

$$\mathrm{QC}_b^A(P,Q) = \sqrt{\sum_{ij} \frac{A_{ij}(p_i - q_i)(p_j - q_j)}{\big(\sum_k A_{ki}(p_k+q_k) \sum_\ell A_{\ell j}(p_\ell + q_\ell)\big)^b}}\,.$$

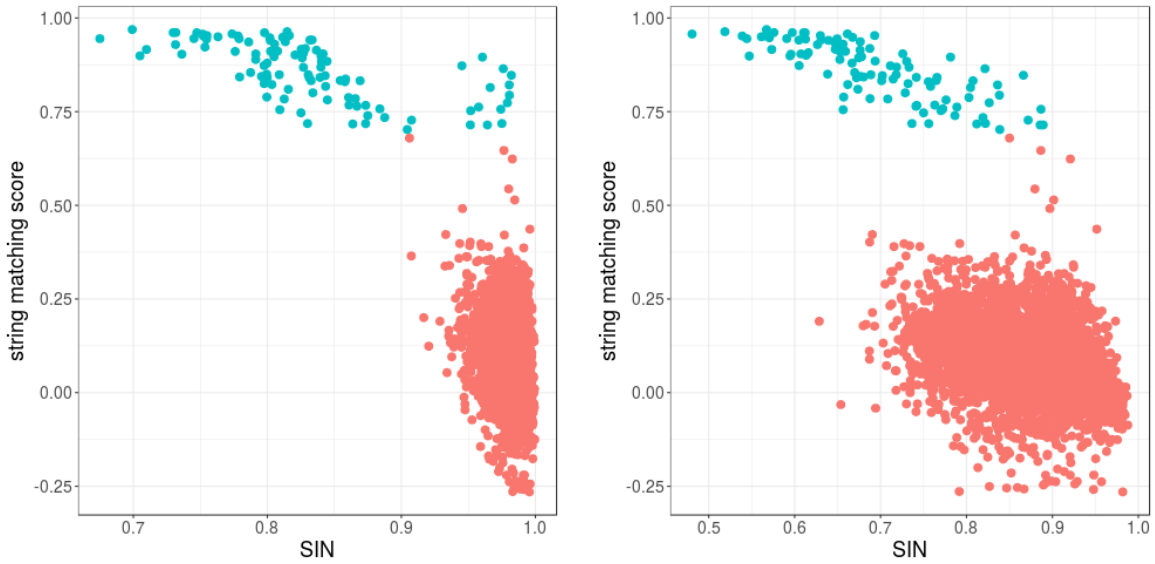
В статье [10] предлагается взять A такой, что $a_{ij} = 1 - d_{ij}/\max_{ij} d_{ij}$, а $b = 0.6$ или 0.9. Однако нет теоретических результатов, говорящий о том, при каких A расстояние QC будет являться метрикой. Требование от расстояния между спектрами быть метрикой не является необходимостью в данной задаче, но его выполнение позволяет ускорить некоторые алгоритмы кластеризации с помощью неравенства треугольника.

В задаче сравнения спектров-гистограмм разница между большими столбцами менее важна, чем разница между маленькими, и важное преимущество QC перед EMD в том, что первое расстояние это учитывает.

Второе преимущество перед EMD — это быстродействие. Алгоритм подсчета QC работает $O(m^2)$, где m — суммарное число ненулевых столбцов гистограмм, но программная реализация предложенная авторами [10] использует разреженность матрицы A и работает значительно быстрее, чем EMD при одном и том же t.

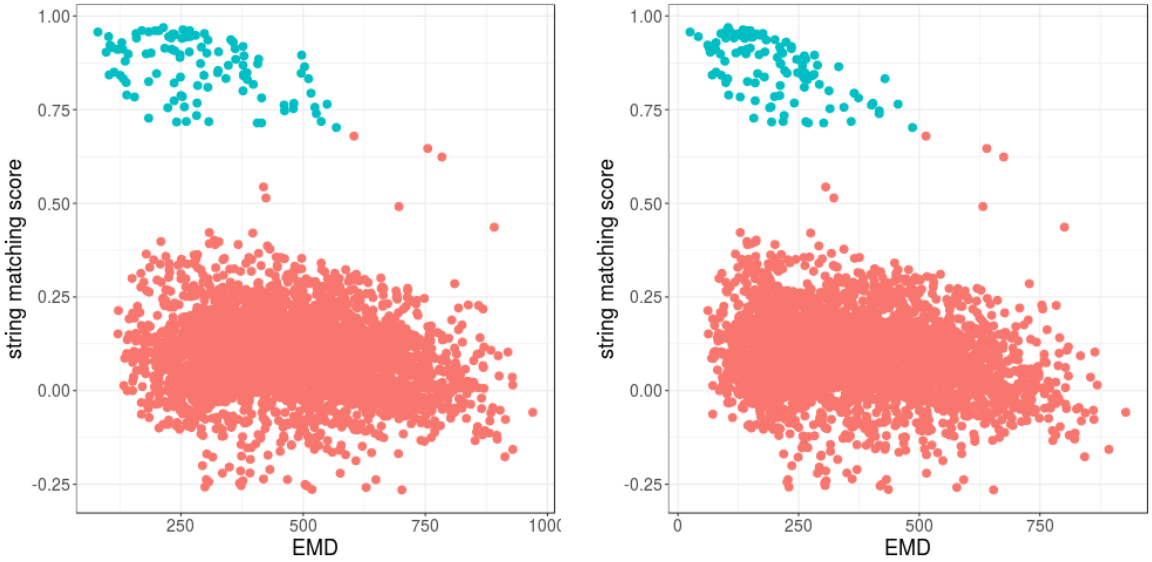
На рисунках 3.3–3.5 представлены графики, показывающие связь между описанными расстояниями между спектрами-гистограммами и оценкой близости пептидов stringMatchingScore. В качестве порога для расстояния d_{ij}, используемого в EMD и QC, было взято $t = 5$, подобранное экспериментально. Каждая точка представляет собой пару «расстояние между гистограммами — оценка близости пептидов», посчитанную для некоторой пары пептидов.

На графиках видно, что по оси абсцисс множества точек хорошо разделяются, и stringMathingScore = 0.7 можно взять оценкой границы разделения близких пептидов от далеких.



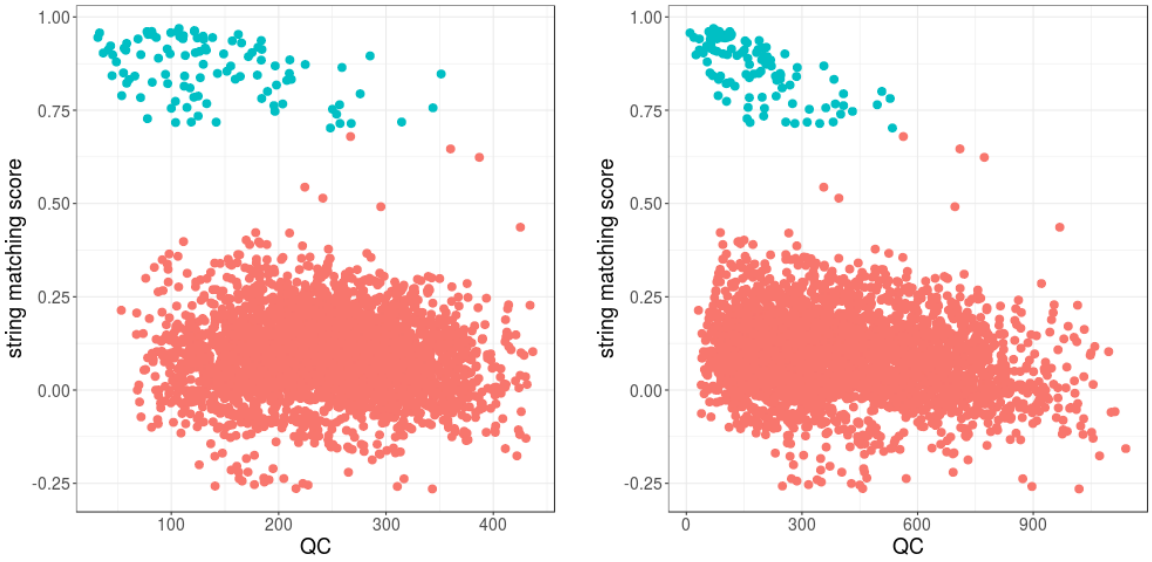
(a) bin length = 5 (b) bin length = 20

Рис. 3.3. Связь синуса между гистограммами с шириной столбцов bin length и stringMathingScore. Синим цветом изображены точки, для которых stringMathingScore > 0.7.



(a) bin length = 5 (b) bin length = 20

Рис. 3.4. Связь метрики EMD между гистограммами с шириной столбцов bin length и stringMathingScore. Синим цветом изображены точки, для которых stringMathingScore > 0.7.



(a) bin length = 5 (b) bin length = 20

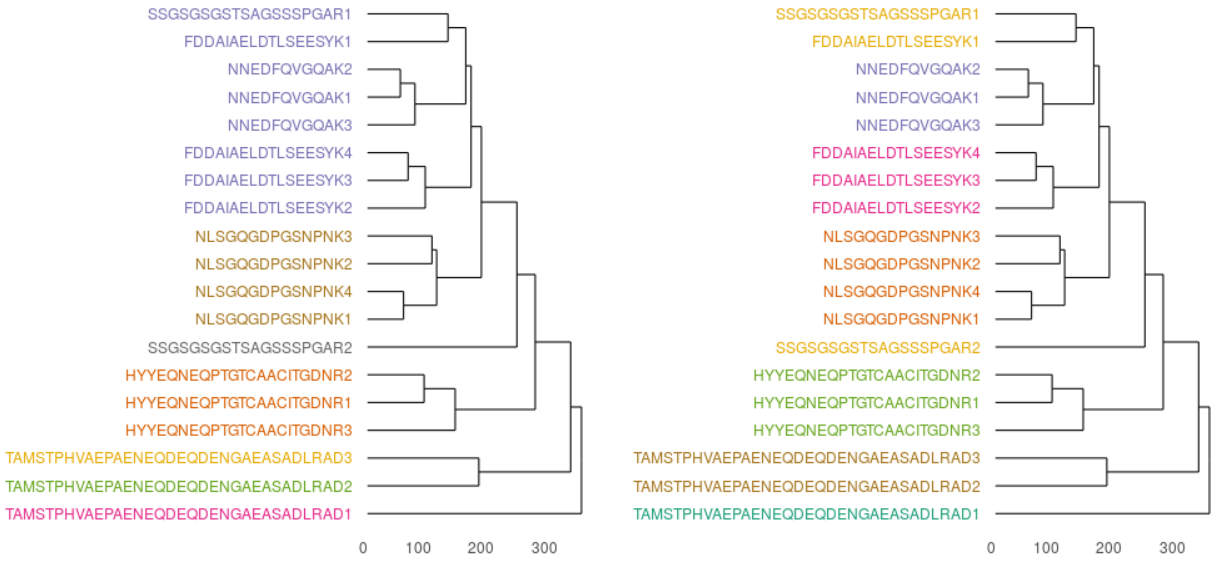
Рис. 3.5. Связь расстояния QC между гистограммами с шириной столбцов bin length и stringMathingScore. Синим цветом изображены точки, для которых stringMathingScore > 0.7.

Также можно заметить, что для bin length = 5 множество точек, для которых stringMathingScore > 0.7, соответствующее далеким друг от друга пептидам, расположено левее, чем в случае bin length = 20. Можно предположить, что с уменьшением bin length — возможно, до некоторого порога — увеличивается его левая граница, и регулируя этот параметр, можно регулировать качество кластеризации каждого из рассмотренных трех расстояний.

3.1.4. Об алгоритме кластеризации

Для кластеризации ожидаемых масс-спектров воспользуемся иерархической кластеризацией. Одно из ее преимуществ по сравнению с другими алгоритмами — это отсутствие требования фиксировать число кластеров, в качестве входа достаточно подать матрицу расстояний между объектами.

Иерархическая кластеризация, реализованная в методе hclust в R, является агломеративной, то есть на каждом шаге кластеры объединяются в более крупные на основе матрицы расстояний. В качестве способа объединения кластеров был выбран метод полной связи (complete linkage). В результате работы алгоритма получаем дендрограмму — дерево, которое отражает связь между кластерами согласно матрице расстояний. Пример дендрограммы показан на рисунках 3.6.



(a) Обрезание на одной высоте

(b) Алгоритм Dynamic Tree Cut

Рис. 3.6. Сравнение методов подрезания дендрограммы, построенной с использованием расстояния QC. Цветом показаны разбиения на кластеры в соответствии с методом подрезания.

Обычно, чтобы получить разбиение данных на кластеры, дендрограмму подрезают на одной высоте, в большинстве случаев такой подход хорошо работает. Однако, если дерево сложной формы с множеством вложенных кластеров, подрезание на одной высоте не приводит к удовлетворительным результатам. Пример такого случая показан на рисунке 3.6(a).

Существует алгоритм, который позволяет правильнее разбить на кластеры, исходя из формы поддеревьев — Dynamic Tree Cut [11].

В статье представлено два подхода — Dynamic Tree, который стоит разбиение используя только дендрограмму, и Dynamic Hybrid, который использует также матрицу расстояний. Опишем схемы обоих подходов.

1. Dynamic Tree.

   Введем вспомогательное определение. Пусть $S = (s_1, \ldots, s_n)$ — веществennозначный вектор. Будет называть точку i точкой перехода, если $s_i > 0$, а $s_{i+1} < 0$. Число последовательных точек j, $j < i$, для которых $\operatorname{sign} s_j = \operatorname{sign} s_i$, будем называть длиной прямой серии и обозначать через $r(i)$. Контрольными точками будем называть $f(i) = i - r(i)$.

   На предварительно построенной дендрограмме алгоритм выполняет подрезание на одной высоте, достаточно большой, например, $0.99 \cdot \max_{ij} d_{ij}$, где $D = \{d_{ij}\}$ — матрица расстояний. Получаем некоторое стартовое разбиение размера m, которое затем разбивается на более мелкие следующим образом. Пусть $\Omega = \mathcal{H}_1, \ldots, \mathcal{H}_m$ — дендрограммы каждого кластера (поддеревья исходной дендрограммы). Затем для каждого кластера запускается функция AdaptiveTreecutCore, которую опишем ниже. Если образовались новые кластеры, то Ω обновляется, затем опять запускается AdaptiveTreecutCore и так до сходимости.

   Метод AdaptiveTreecutCore связан с методом TreecutCore. На вход TreecutCore подается дендрограмма в виде последовательности высот $\mathcal{H} = (h_1, \ldots, h_n)$, высота ℓ и порог τ. В первую очередь считается последовательности $\widehat{\mathcal{H}} = (h_1 - \ell, \ldots, h_n - \ell)$ и ее контрольные точки $\{f(i)\}_{i=1}^n$, из них отбираются те $f(i)$, для которых $r(i) > \tau$. Две последовательные $f(i)$ для которых выполняется это условие задают кластер. Получаем разбиение исходного множества на кластеры.

   В AdaptiveTreecutCore метод TreecutCore вызывается для трех значений высот $\ell_m = \frac{1}{n}\sum_{i=1}^n h_i$, $l_u = \frac{1}{2}(\ell_m + \max\{h_1, \ldots, h_n\})$, $l_d = \frac{1}{2}(\ell_m + \max\{h_1, \ldots, h_n\})$ — сначала для ℓ_m, и если множество не разбилось хотя бы на 2 кластера, то запускается на ℓ_u, и если разбиения опять не произошло, то на ℓ_d. Цель этого адаптивного подхода в том, чтобы правильно обработать ситуацию, когда высоты в каком-нибудь поддереве $\mathcal{H}$ мало отличаются от ℓ_m.

2. Dynamic Hybrid. Этот алгоритм устроен сложнее, поэтому опишем его поверхностно. Алгоритм выполняется в два шага. На первом шаге с помощью четырех критериев формы поддеревьев выделяются некоторые кластеры.

   - Первый критерий — это ограничение на количество объектов в кластере.
   - Второй критерий удаляет из кластера объекты, если они находятся достаточно далеко от оставшихся, даже если они принадлежат одной ветке дендрограммы.
   - Согласно третьему критерию, каждый кластер должен быть отделен от своего окружения некоторым зазором, длина которого регулируется пользователем.
   - По четвертому критерию ядро кластера должно быть плотным, где под ядром понимаются объекты, расположенные на нижних уровнях дендрограммы

данного кластера.

На втором шаге объекты, которые не попали ни в один кластер, пытаются распределить по существующим кластерам. Реализованный метод похож на модифицированный метод Partitioning Around Medoids (PAM, [12]).

Будем использовать подход Dynamic Hybrid, так как он дает более точные результаты.

3.1.5. Описание эксперимента с реальными данными

Ранее в главе были предложены три способа измерять близость спектров как гистограмм и продемонстрировано, что оптимальная ширина столбца гистограммы должна быть небольшой. В виду того, что на практике метрика EMD считает значительно медленнее двух остальных величин при небольших bin length, то далее мы остановимся на рассмотрении расстояний SIN и QC.

Цель данного эксперимента — выявить различия между этими расстояниями и определить оптимальное значение bin length.

Рассмотрим набор масс-спектров, для каждого которых с некоторой точностью был идентифициварон пептид. Нас будут интересовать сами строки-пептиды, а масс-спектры использоваться в экспериментах не будут. Количество уникальных строк-пептидов $M = 17000$, обозначим это множество через $\mathbf{P}$. С помощью метода calculateFragments из пакета MSnbase в R для каждого пептида построим ожидаемый спектр.

В виду того, что вычислять оценку близости для каждой пары пептидов из данных M требует $M(M-1)$ вызовов функции stringMatchingScore, будем рассматривать K выборок небольшого размера из этих данных. Это позволит определить свойства каждого расстояния между спектрами и их связь с близостью строк-пептидов без весомых временных затрат.

Построим K непересекающихся выборок так, как описано ниже. Введем множество пептидов $\mathbf{P}_j = \cup_{i=1}^{j} \mathbf{S}_i$, то есть те пептиды, которые уже включены во все построенные выборки на j-м шаге, $P_0 = \varnothing$. Фиксируем число ожидаемых кластеров k в каждой выборке $\mathbf{S}_i$. Для $j = 1, \ldots, K$:

1. Строим выборку размера $M/4$ из множества $\mathbf{P} \setminus \mathbf{P}_{j-1}$.
2. Кластеризуем пептиды из этой выборки с помощью иерархической кластеризации по матрице из stringMatchingScore так, что в каждом кластере stringMatchingScore > 0.7.
3. Удаляем кластеры размера < 3, случайно выбираем k кластеров, из них составляем выборку пептидов $\mathbf{S}_j$.

Граница на stringMatchingScore взята, исходя из формы областей точек на графиках 3.3–3.5, размер выборки $M/4$ позволяет быстро сделать вышеописанные вычисления. Кластеры размера 2 и меньше удаляются с целью адекватного подсчета характеристик кластеризации.

Почему именно так устроена процедура построения выборок? Конечно, можно было изначально разделить M пептидов на равные непересекающиеся множества, но тогда сложно подобрать параметры, чтобы получить большого размера выборки и оптимальное их количество. Если же строить j-ю выборку по всем пептидам $\mathbf{P}$, а не по $\mathbf{P}_j$, то выборки будут сильно пересекаться, и мы не учтем значительную часть данных. Вышеописанная процедура — это объединение этих двух подходов, чтобы компенсировать недостатки обоих.

Для каждой выборки $\{\mathbf{S}_j\}_{j=1}^K$ и описанных выше расстояний SIN и EMD:

1. Вычисляем матрицу расстояний.
2. Запускаем иерархическую кластеризацию с динамическим обрезанием дерева.
3. Считаем всевозможные статистики полученного разбиения.

Статистики затем усредняются по выборкам.

Для эксперимента были выбраны $K = 8$ и $k = 35$. Размеры выборок при таких параметрах варьировались от 110 до 140. В процессе генерации выборок было замечено, что на первой итерации среди $M/4 = 4250$ пептидов только около 250 пептидов формируют кластеры размера больше двух. Это позволяет сделать вывод, что большую часть данных занимают пептиды, непохожие друг на друга в смысле расстояния между строками.

Были рассмотрены три характеристики качества кластеризации и построены графики их зависимости от bin length в сетке $\{0.5, 1, 2, 5, 7, 10, 12, 15, 17, 20\}$. Описание всех трех характеристик приведено ниже.

1. Средняя по кластерам минимальная оценка близости между пептидами из одного кластера.

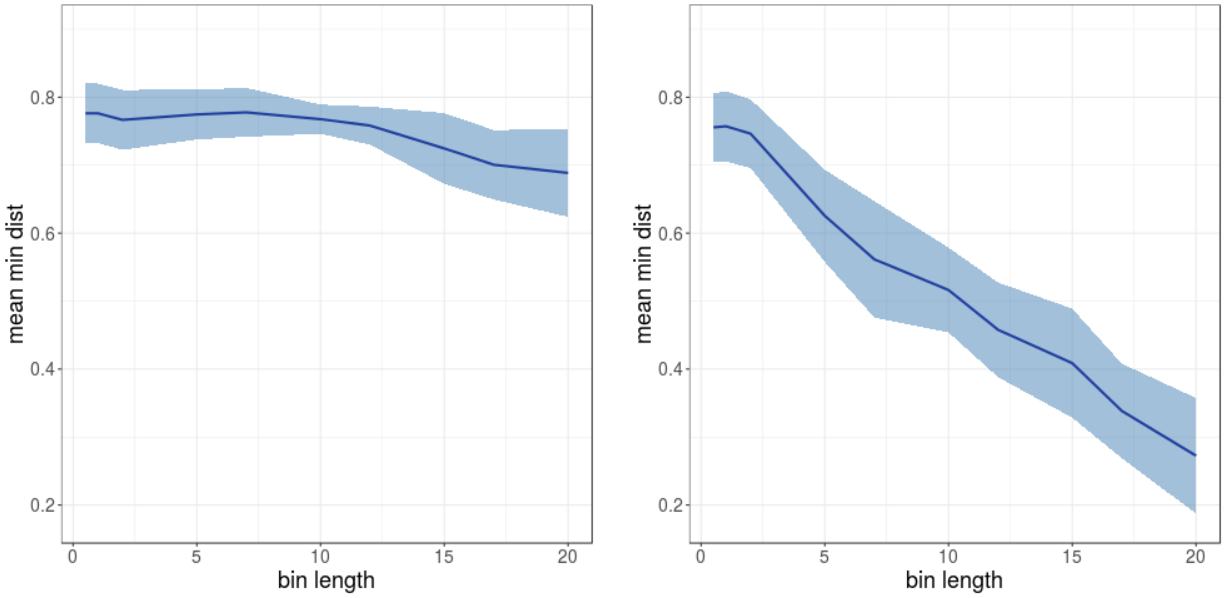
   Цель этой характеристики показать, как зависит минимальная в кластере оценка близости между пептидами от bin length (в среднем по всем кластерам). Чем она выше, тем больше кластеров, в которых stringMatchingScore > 0.7, то есть состоящих из достаточно близких пептидов.

   На рисунке 3.7 видно, что для обоих расстояний эта характеристика возрастает с уменьшением bin length. Максимальное значение статистики находится между 0.7 и 0.8, для расстояния SIN статистика находится в этом диапазоне при значениях bin length от 0.5 до 12, для QC — в диапазоне от 0.5 до 1 включительно.

2. Доля правильно кластеризованных пептидов.

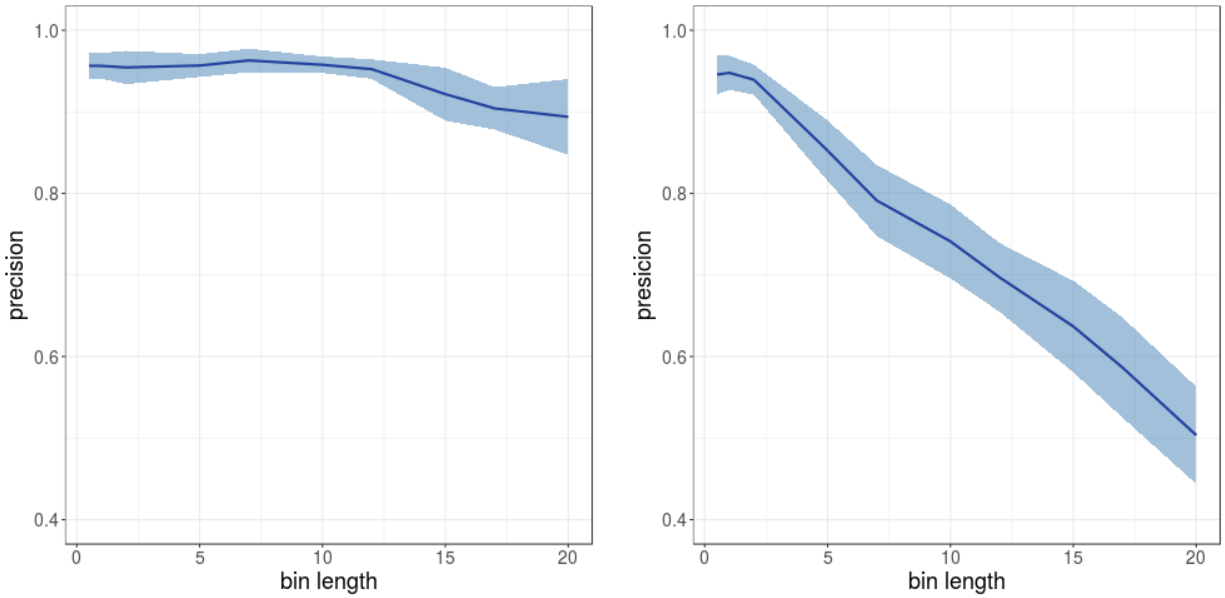
   Она считается как 1−«доля пептидов, которые лежат в кластере с пептидами, для которых их stringMatchingScore < 0.7». Аналог точности (precision) в классификации.

   Как показано на рисунке 3.8, для расстояния SIN характеристика больше 0.9 при bin length в диапазоне 0.5 до 12, а QC — при bin length в диапазоне 0.5 до 1 включительно.



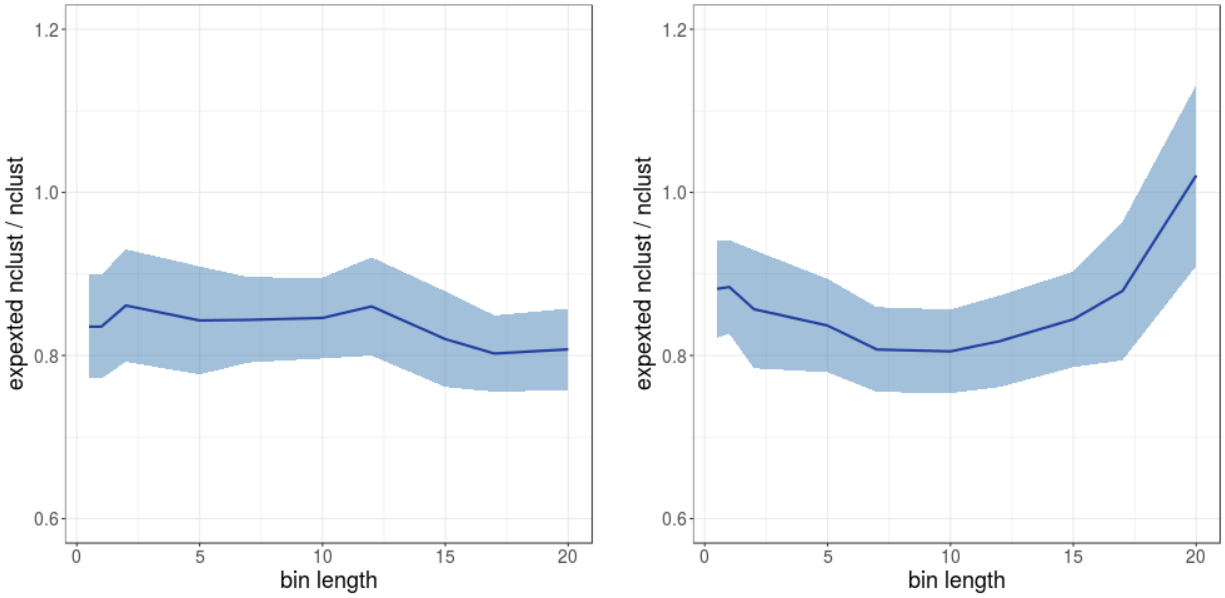
(a) Расстояние SIN (b) Расстояние QC

Рис. 3.7. Зависимость средней по кластерам минимальной оценки близости между пептидами из одного кластера от bin length.



(a) Расстояние SIN (b) Расстояние QC

Рис. 3.8. Зависимость доли правильно кластеризованных пептидов от bin length.



(a) Расстояние SIN

(b) Расстояние QC

Рис. 3.9. Зависимость отношения ожидаемого числа кластеров к фактическому от bin length.

3. Отношение ожидаемого числа кластеров к фактическому, то есть к количеству кластеров, полученных с помощью кластеризации по матрице расстояний между масс-спектрами.

   На рисунке 3.9 изображена эта статистика в зависимости от ширины столбца гистограммы для расстояний SIN и QC. Для bin length $\leq$ 15 статистика в обоих случаях меньше единицы, значит, число фактических кластеров превышает ожидаемое — получаем более мелкое разбиение.

   Для расстояния QC статистика убывает с ростом bin length на промежутке от 0.1 до 10, а затем возрастает.

   При значениях bin length 0.1 до 5 статистика для обоих расстояний находится в похожем диапазоне от 0.75 до 0.95.

На основе этих графиков можно сделать вывод, что оба расстояния дают подходящие результаты кластеризации при bin length = 1 и близких к этому значению, но диапазон bin length, для которых выше рассмотренные статистики дают оптимальные результаты при расстоянии SIN, от 0.1 до 12, в отличие от QC, у которого оптимальные bin length находятся в промежутке от 0.1 до 1 включительно. Чем больше bin length, тем меньше памяти занимает представление спектра в виде гистограммы, и в этом смысле использование расстояния SIN оптимальнее, чем QC. Кроме того, для подсчета первого требуется гарантировано линейное время, а для подсчета второго — квадратичное в худшем случае.

Заключение

Масс-спектрометрия — широко используемый инструмент определения химического состава веществ. В этой работе были рассмотрены две задачи, связанные с анализом масс-спектров.

Первая задача связана с фильтрацией пиков в масс-спектрах. Исходно масс-спектры зашумлены и могут включать тренд, это влияет на результаты дальнейшей обработки. Текущая фильтрация пиков в Дерепликаторе имеет ряд недостатков, поэтому были исследованы актуальные методы фильтрации пиков PROcess и MassSpecWavelet из пакета Bioconductor в R. Оба алгоритма оценивают некоторым образом отношение сигнала к шуму (SNR) в каждой точке спектра, а затем отбираются те пики, для которых SNR больше некоторого порога. Основное отличие этих двух алгоритмов в том, что в PROcess существуют отдельные шаги для извлечения оценок шума и тренда, а в MassSpecWavelet в виду особенностей использования непрерывного вейвлет-преобразования это не требуется.

В качестве альтернативы отбора пиков в Дерепликаторе был выбран MassSpecWavelet, но его алгоритм неявно предполагает, что m/z в спектре равноотстоящие. В ProteoWizard реализована модификация MassSpecWavelet, которую можно применять для не равноотстоящих m/z, однако она нуждалась в доработке — чтобы правильно обрабатывались спектры небольшой длины, все относительные параметры были переведены в абсолютные. Измененная реализация была протестистирована на масс-спектрах с известными пептидами, чтобы убедиться в корректности.

Вторая задача относится к ускорению процедуры идентификации пептидов. Существуют методы, позволяющие определить химическую формулу пептида по его масс-спектру, один из них использует для этих целей базу данных пептидов. Чтобы ускорить поиск подходящего пептида в базе данных, а точнее ожидаемого спектра, который строится по каждому пептиду из базу данных, предложено кластеризовать эмпирические масс-спектры, для которых необходимо узнать пептид, и ожидаемые спектры.

Задача кластеризации первых намного сложнее ввиду зашумленности спектров и ряда других причин, поэтому в работе исследовались ожидаемые спектры. В качестве оптимального векторного вложения ожидаемых спектров было выбрано представление в виде гистограммы, а для кластеризации выбран алгоритм иерархической кластеризации с динамическим обрезанием дерева. На реальных данных были протестированы три варианта расстояний между спектрами, и два из них — синусное расстояние (SIN) и Quadratic-Chi Histogram Distance (QC) — показали подходящие результаты. Использование расстояния SIN имеет ряд преимуществ перед QC. Оно вычисляется гарантировано за линейное время, и максимальная ширина гисторамммы, при которой использование расстояние SIN приводит к приемлемому качеству кластеризации, больше, чем для QC, что позволяет экономнее использовать память для представления спектров.

Список литературы

1. Mohimani H., Gurevich A. et al. Dereplication of Peptidic Natural Products Through Database Search of Mass Spectra // Nature Chemical Biology. — 2016. — Vol. 13, no. 1. — P. 30–37.
2. Yang C., He Z., Yu W. Comparison of public peak detection algorithms for MALDI mass spectrometry data analysis // BMC Bioinformatics. — 2009. — Vol. 10. — P. 4–16.
3. Li X. — PROcess: Ciphergen SELDI-TOF Processing, 2005. — R package version 1.48.0.
4. Du P., Kibbe W. A., Lin S. M. Improved peak detection in mass spectrum by incorporating continuous wavelet transform-based pattern matching // Bioinformatics. — 2006. — Vol. 22, no. 17. — P. 2059–2065.
5. French W. R., Zimmerman L. J. et al. Wavelet-Based Peak Detection and a New Charge Inference Procedure for MS/MS Implemented in ProteoWizard's msConvert // Journal of Proteome Research. — 2015. — Vol. 14, no. 2. — P. 1299–1307.
6. Henikoff J. G., Henikoff S. Amino acid substitution matrices from protein blocks // Proceedings of the National Academy of Sciences of the United States of America. — 1992. — Vol. 89, no. 22. — P. 10915–10919.
7. Rubner Y., Tomasi C., Guibas L. J. The Earth Mover's Distance as a Metric for Image Retrieval // International Journal of Computer Vision. — 2000. — Vol. 40, no. 2. — P. 99–121.
8. Pele O., Werman M. A Linear Time Histogram Metric for Improved SIFT Matching // Computer Vision – ECCV 2008: 10th European Conference on Computer Vision, Marseille, France, October 12-18, 2008, Proceedings, Part III. — Springer Berlin Heidelberg, 2008. — P. 495–508.
9. Pele O., Werman M. Fast and robust Earth Mover's Distances // 2009 IEEE 12th International Conference on Computer Vision. — 2009. — P. 460–467.
10. Pele O., Werman M. The Quadratic-Chi Histogram Distance Family // ECCV. — 2010. — P. 749–762.
11. Langfelder P., Zhang B., Horvath S. Defining clusters from a hierarchical cluster tree: the Dynamic Tree Cut package for R // Bioinformatics. — 2008. — Vol. 24, no. 5. — P. 719–720.
12. Park H. S., Jun C. H. A simple and fast algorithm for K-medoids clustering // Expert Systems with Applications. — 2009. — Vol. 36, no. 2. — P. 3336–3341.